

IN THE CLAIMS:

This listing of claims replaces all prior versions and listings of claims in the application.

- 1 1. (Currently Amended) A monitoring ~~debug~~ interface, comprising:
2 logic responsive to a pre-fork event, the pre-fork event responsive to a vfork
3 system fork instruction call, wherein the pre-fork event includes indicia that identifies
4 a child process to be created in accordance with the vfork system fork instruction call.
- 1 2. (Original) The interface of claim 1, wherein the indicia comprises a
2 process identifier.
- 1 3. (Currently Amended) The interface of claim 1, wherein the pre-fork
2 event is delivered ~~by a parent process~~ to a software monitor.
- 1 4. (Original) The interface of claim 3, wherein the parent process was
2 under run-time analysis by the software monitor.
- 1 5. (Currently Amended) The interface of claim 3, wherein the pre-fork
2 event is delivered before the vfork system call is executed by the parent process ~~parent~~
3 ~~process was instrumented for run-time analysis by the software monitor.~~
- 1 6. (Currently Amended) The interface of claim 3, wherein the software
2 monitor responds by executing the child process until completion ~~pre-fork event is~~
3 ~~delivered before the fork instruction call is executed by the parent process.~~
- 1 7. (Currently Amended) The interface of claim 3 ~~6~~, wherein the software
2 monitor responds to indicia of completion of the child process by resuming execution
3 of the parent ~~executing the child~~ process until completion.

1 8. (Currently Amended) The interface of claim 7, wherein the software
2 monitor ensures that the first event pertaining to the parent process and received after
3 completion of the child process is an event denoting completion of a vfork system call
4 ~~responds to indicia of completion of the child process by executing the parent process~~
5 ~~until completion.~~

1 9. (Canceled) The interface of claim 8, wherein the software monitor
2 ensures that the first event in the parent process that spawned the child process is the
3 fork event itself.

1 10. (Currently Amended) A method for controlling the execution of a child
2 process created from a parent process, wherein the parent process is instrumented by a
3 software tool, the method comprising ~~the steps of:~~
4 receiving indicia that a vfork system call ~~fork instruction~~ will be executed by
5 the parent process;
6 suspending execution of the parent process;
7 extracting a process identifier from the indicia of the vfork system call ~~fork~~
8 ~~instruction~~, the process identifier corresponding to a child process to be generated by
9 the parent process when the parent process executes the vfork system call ~~fork~~
10 ~~instruction~~;
11 setting a process monitor thread to observe the child process; and
12 resuming execution of the parent process to enable the parent process to
13 execute the vfork system call ~~fork instruction~~.

1 11. (Currently Amended) The method of claim 10, further comprising:
2 waiting for indicia that the child process has invoked at least one of an exec
3 system call and an _exit system call or has been terminated by an operating system
4 ~~nominally terminated~~; and
5 setting a process monitor thread to observe the parent process.

1 12. (Original) The method of claim 11, wherein setting a process monitor
2 thread comprises enabling observation of trace events generated by the parent process.

1 13. (Currently Amended) The method of claim 10, wherein receiving
2 indicia comprises receiving a pre-fork event.

1 14. (Original) The method of claim 13, wherein the pre-fork event includes
2 the process identifier.

1 15. (Original) The method of claim 10, wherein setting a process monitor
2 thread comprises enabling observation of trace events generated by the child process.

1 16. (Currently Amended) A method for executing a parent process
2 monitored ~~instrumented~~ by a software tool to ensure execution of a child process when
3 the parent process contains a vfork system call ~~fork instruction~~, the method comprising:
4 determining if a vfork system call ~~fork instruction~~ is about to be executed;
5 generating a pre-fork event that includes indicia of a child process that will be
6 generated by the vfork system call ~~fork instruction~~;
7 sending the pre-fork event to the software tool;
8 waiting for indicia that the software tool successfully processed the pre-fork
9 event;
10 executing the vfork system call ~~fork instruction~~; and
11 suspending execution of the parent process.

1 17. (Currently Amended) The method of claim 16, further comprising:
2 waiting for indicia that the child process has invoked at least one of an exec
3 system call and an _exit system call or has been terminated by an operating system;
4 and
5 resuming execution of the parent process.

1 18. (Original) The method of claim 17, wherein waiting for indicia that the
2 child process has terminated comprises a trace event.

1 19. (Original) The method of claim 16, wherein indicia of the child process
2 comprises a process identifier.

1 20-28. (Canceled)

1 29. (New) A method for controllably switching a target process of a
2 process monitor thread between an instrumented parent process and a child process
3 generated by the parent process, the method comprising:
4 checking whether the successful initiation of the child process can be asserted;
5 when the successful initiation of the child process cannot be asserted,
6 checking if the parent process responsible for creating the child process
7 received indicia of a failure of a vfork system call designated to create the child
8 process;
9 when the indicia has not been received,
10 waiting an amount of time before rechecking for the successful initiation of the
11 child process; otherwise,
12 notifying a software monitor of the unsuccessful initiation of the child process
13 and resuming execution of the parent process;
14 monitoring the parent process;
15 otherwise, when the successful initiation of the child process can be
16 asserted,
17 monitoring the successfully created child process.

1 30. (New) The method of claim 29, wherein the step of checking whether
2 the successful initiation of the child process can be asserted comprises verifying the
3 success of a trace event by using the process identifier of the child process.

1 31. (New) The method of claim 29, wherein checking if a parent process
2 responsible for creating the child process received indicia of a failure comprises
3 searching for a trace event while performing a non-blocking trace wait on the parent
4 process.

1 32. (New) The method of claim 29, further comprising:
2 aborting child process monitoring when the initiation of the child process is
3 unsuccessful.

1 33. (New) An operating system, comprising:
2 a pre-fork event, the pre-fork event responsive to a vfork system call wherein
3 the pre-fork event includes indicia that identifies a child process to be created in
4 accordance with the vfork system call.

1 34. (New) The operating system of claim 33, wherein the indicia
2 comprises a process identifier.

1 35. (New) A computer readable medium, comprising:
2 logic responsive to a pre-fork event, the pre-fork event responsive to a vfork
3 system call wherein the pre-fork event includes indicia that identifies a child process
4 to be created in accordance with the vfork system call.

1 36. (New) The computer readable medium of claim 35, wherein the
2 indicia comprises a process identifier.